Node Linking

1 min

Often, due to the data structure, nodes may only be linked to from a single other node. This makes it very important to consider how you implement modifying or removing nodes from a data structure.

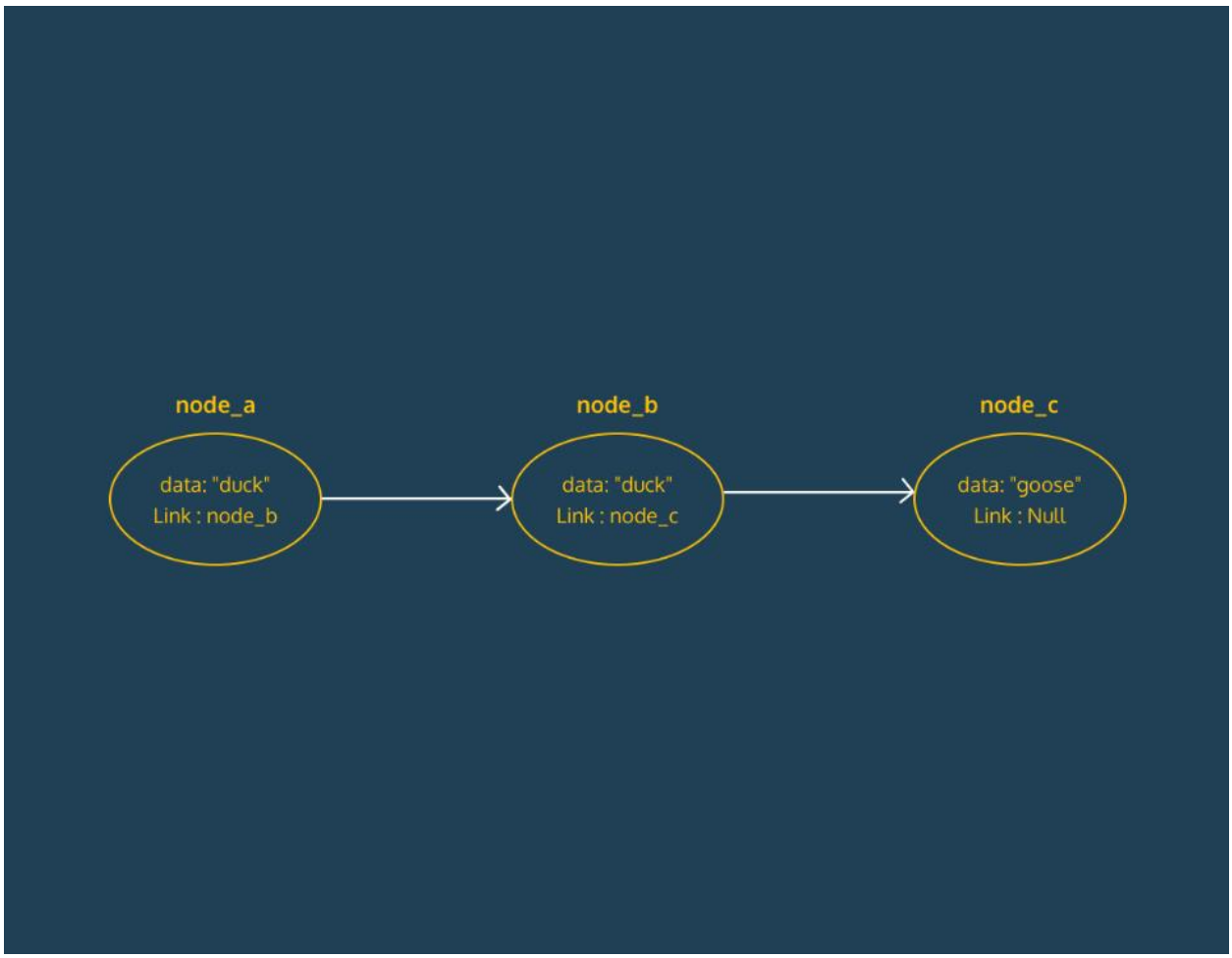
If you inadvertently remove the single link to a node, that node's data and any linked nodes could be "lost" to your application. When this happens to a node, it is called an *orphaned* node.

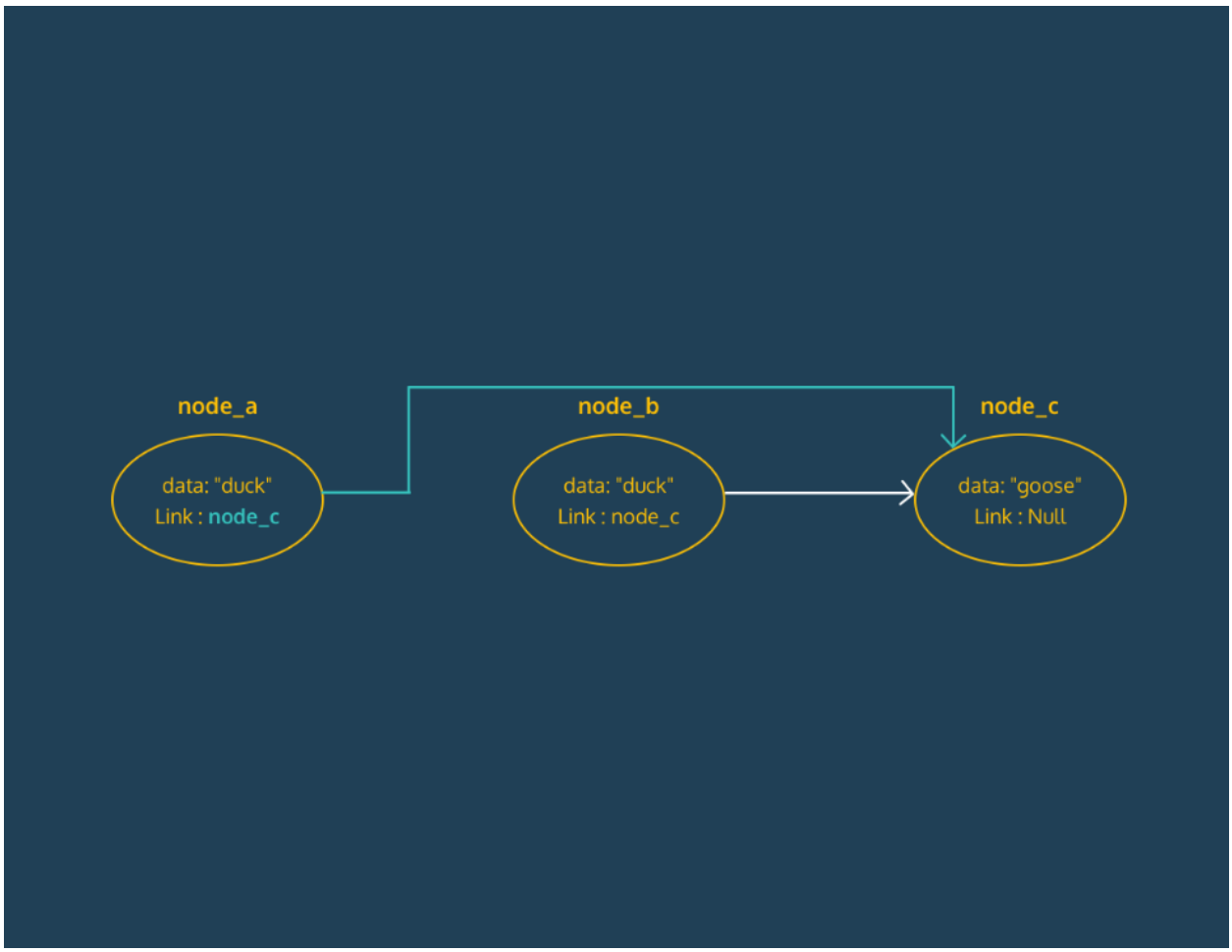
Examine the nodes in the diagram. `node_c` is only linked to by `node_b`. If you would like to remove `node_b` but not `node_c`, you can't simply delete the link from `node_a` to `node_b`.

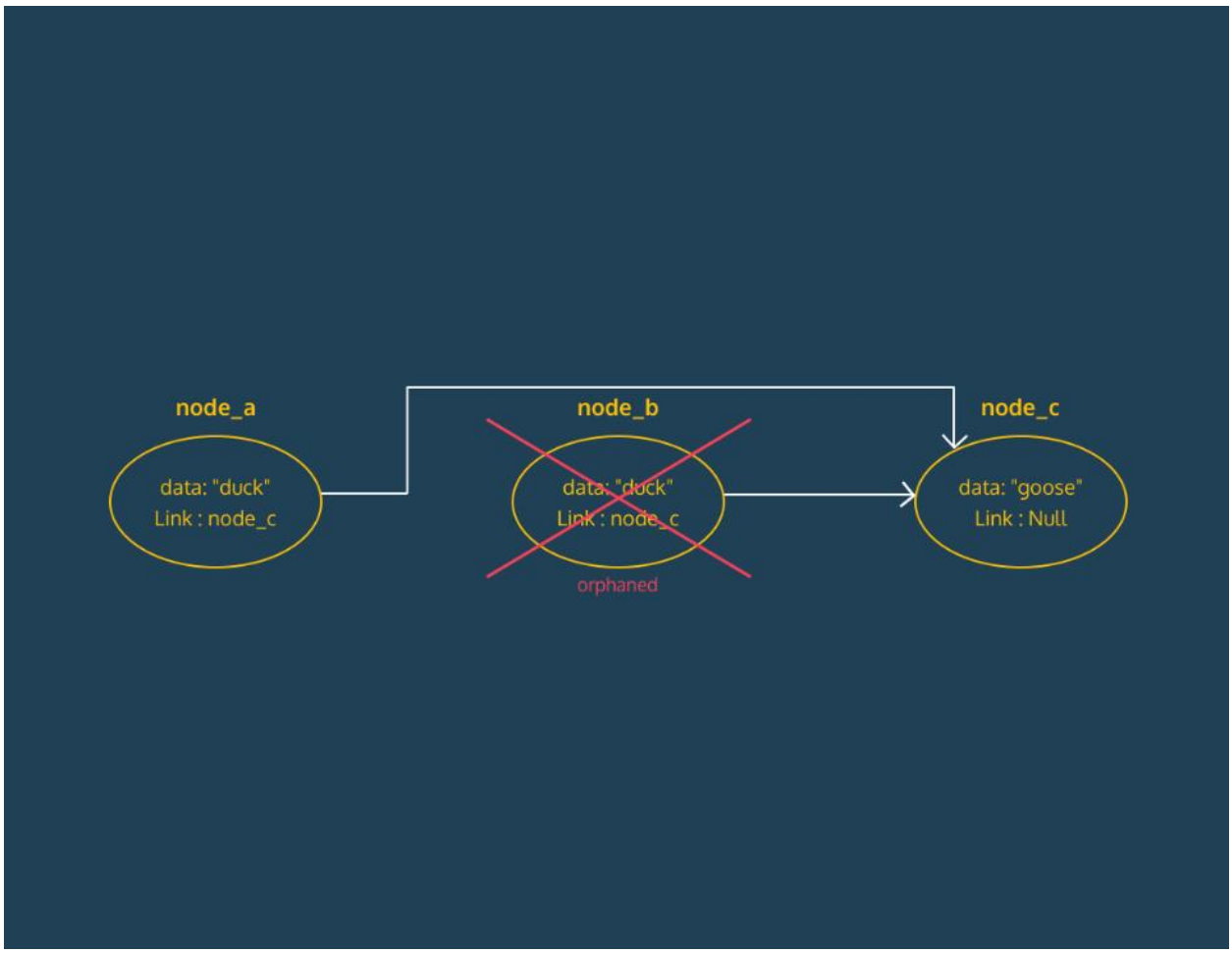
The most straightforward method to preserve `node_c` would be to change the link in `node_a` to point to `node_c` instead of `node_b`. However, some data structures may handle this in a different manner.

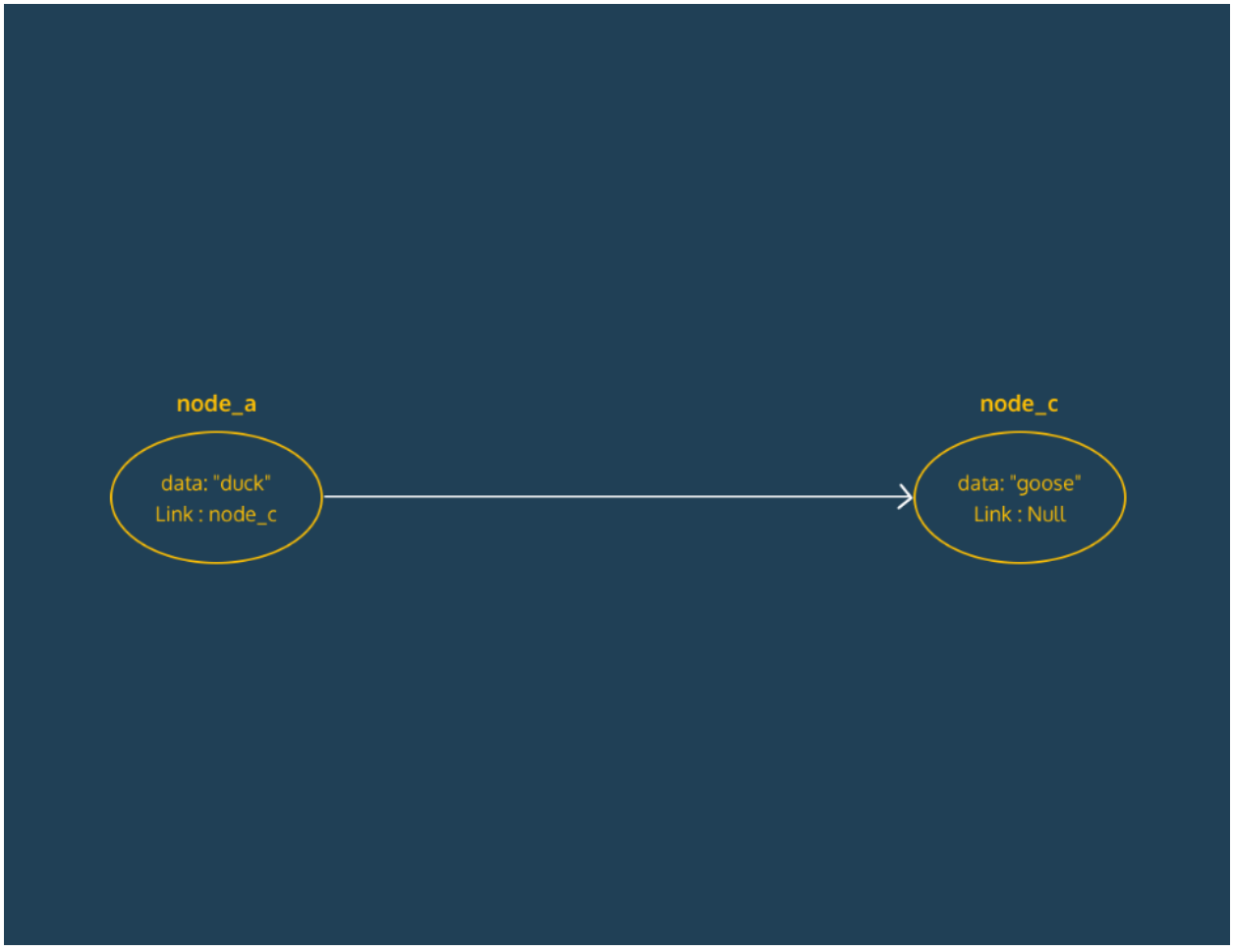
Instructions

Is it necessary to "delete" `node_b` if we change `node_a` to reference `node_c` instead?


node_a
data: "duck"
Link : node_b
node_b
data: "duck"
Link : node_c
node_c
data: "goose"
Link : Null


node_a
node_b
node_c
data: "duck"
Link : node_c
data: "duck"
Link : node_c
data: "goose"
Link : Null


node_a
data: "duck"
Link : node_c
node_b
data: "duck"
Link : node_c
orphaned
node_c
data: "goose"
Link : Null


node_a
data: "duck"
Link : node_c
node_c
data: "goose"
Link : Null